User-Controlled Creation of Multiresolution Meshes

Erik Pojar, Rockstar Vienna

Dieter Schmalstieg, Technical University of Vienna

Outline

- Motivation, Goals
- Weighted Error Metric
- User Interface
- Results, Demo
- Conclusion, Future Work

Motivation(1)

Most existing methods for mesh simplification

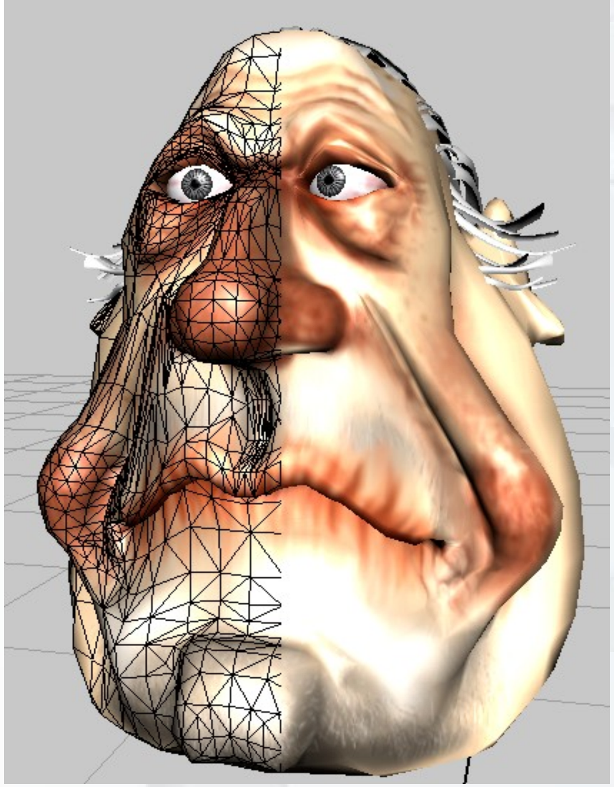
- focus on automatic simplification
- offer only little control for the user
- fail to account for functional and semantic importance of mesh regions

Motivation(2)

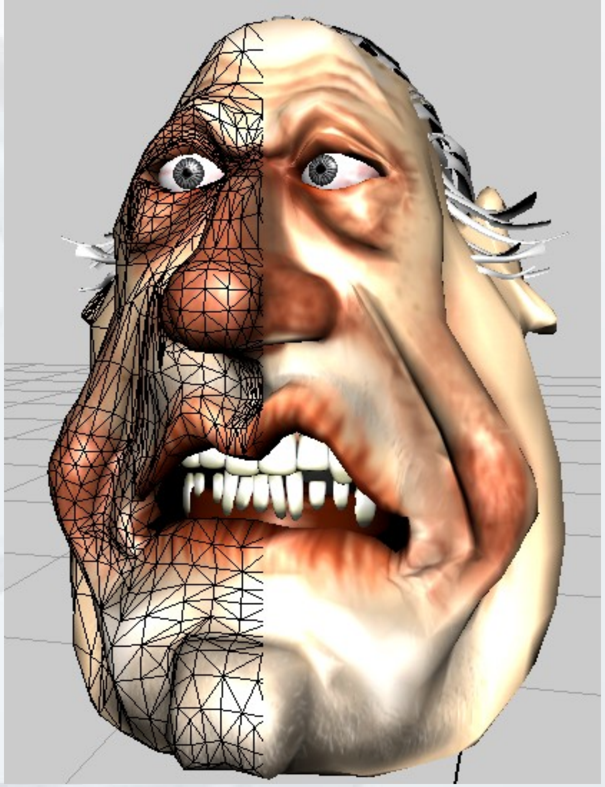
Functional importance arises when a model is put to uses that cannot be deferred from its geometry alone.

For example, the deformable regions of an animated model usually need higher resolution than rigid regions.

Motivation(3)



Rest pose



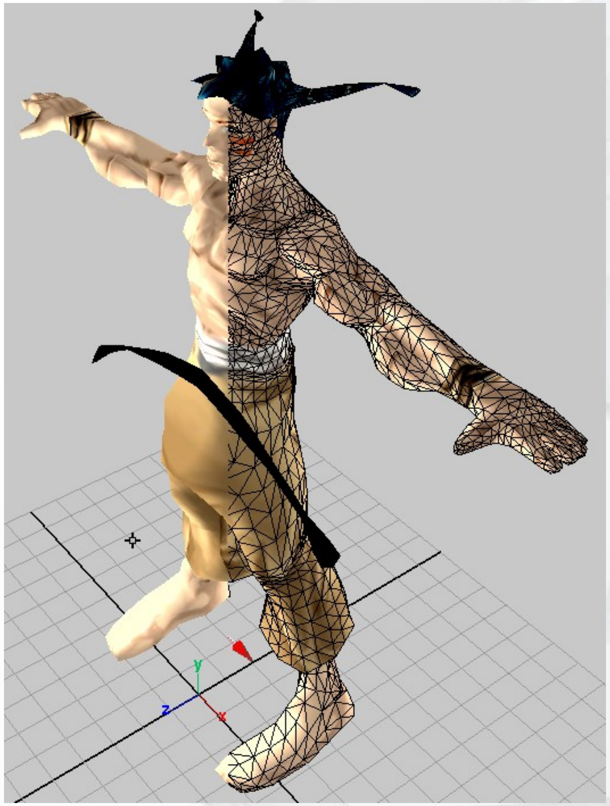
Animated

Motivation(4)

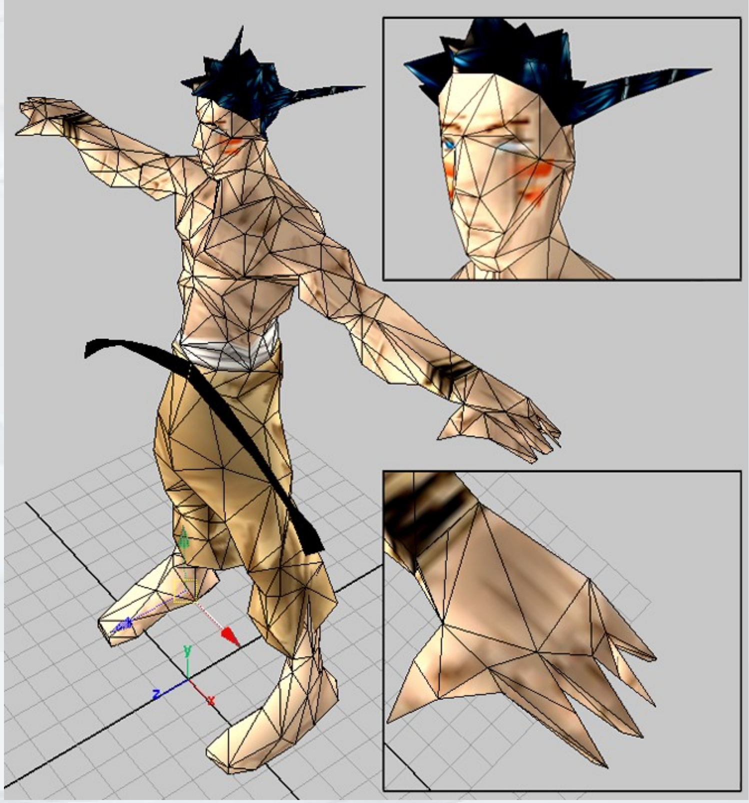
- ***Semantic importance*** **is connected to features of a model that have higher perceptional importance than others.**

- **For example, features such as eyes in a face are semantically crucial but geometrically small.**

Motivation(5)



Original



Simplified to 15%

Goals

- Give users more control over mesh simplification process
- Create a highly useful, intuitively to use tool
- Bridge the gap between academic research and real-world use

Quadric Error Metric

By Michael Garland and Paul Heckbert (1997)

(Using attribute preserving metric by Hoppe)

- Simplification through repeated edge contractions
- QEM assigns both cost and optimal vertex position to each edge
- Edges of low cost are contracted first

Weighted Error Metric(1)

The simplification is clearly driven by the cost of the contractions.

→ Idea: let the user change the cost of contraction and thus control the simplification.

Weighted Error Metric(2)

The cost of contracting an edge (v_i, v_j) is weighted by a user-controlled value $\omega(i, j)$:

$$\mathrm{cost}_w = \omega(i, j) \cdot \mathrm{cost}_g\,(i, j)$$

Where cost_g is the geometric cost of the contraction, the original quadric cost

Weighted Error Metric(3)

The weight ω(i, j) is kept separate from the quadric Q at all times, only post-multiplied.

→ Geometric properties of the contraction are unchanged

→ Weights can be propagated more freely

Weighted Error Metric(4)

The cost function ω is defined through vertex weights w_i:

- **Each vertex v_i is assigned a weight $w_i \geq 0$**
- **$\omega(i, j)$ = average(w_i, w_j)** ***- or -***

  min(w_i, w_j) ***- or -***

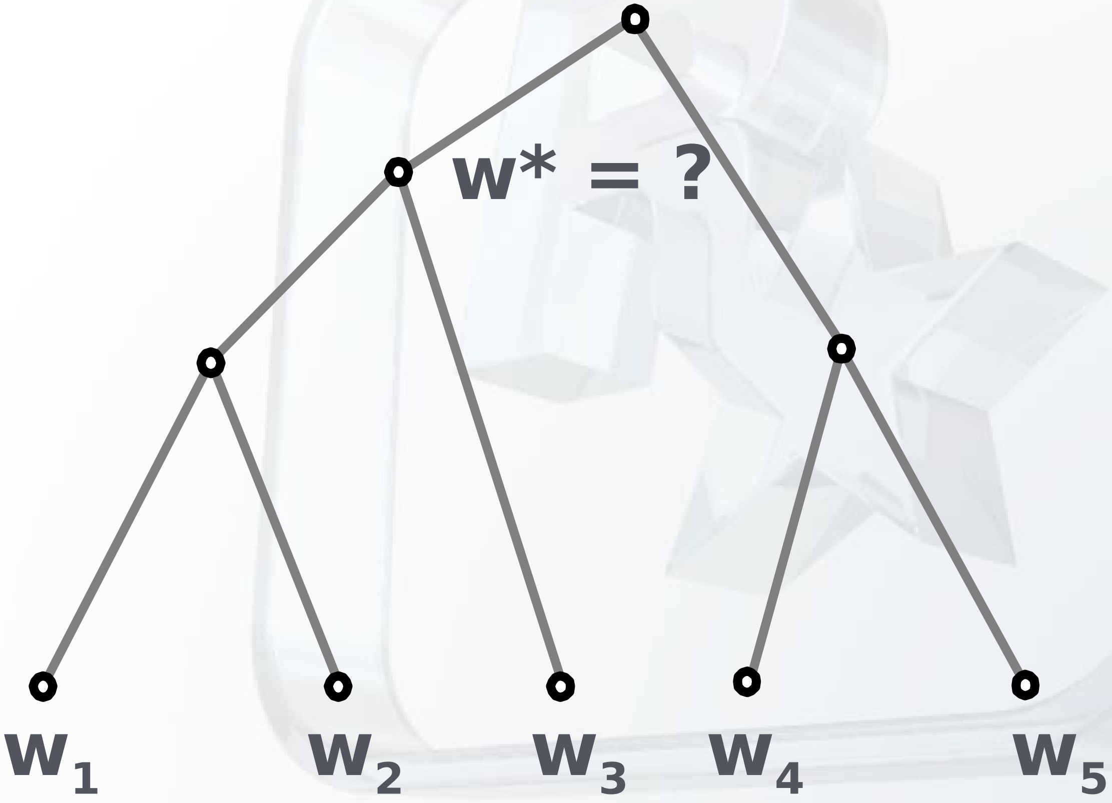
  max(w_i, w_j)

Weighted Error Metric(5)

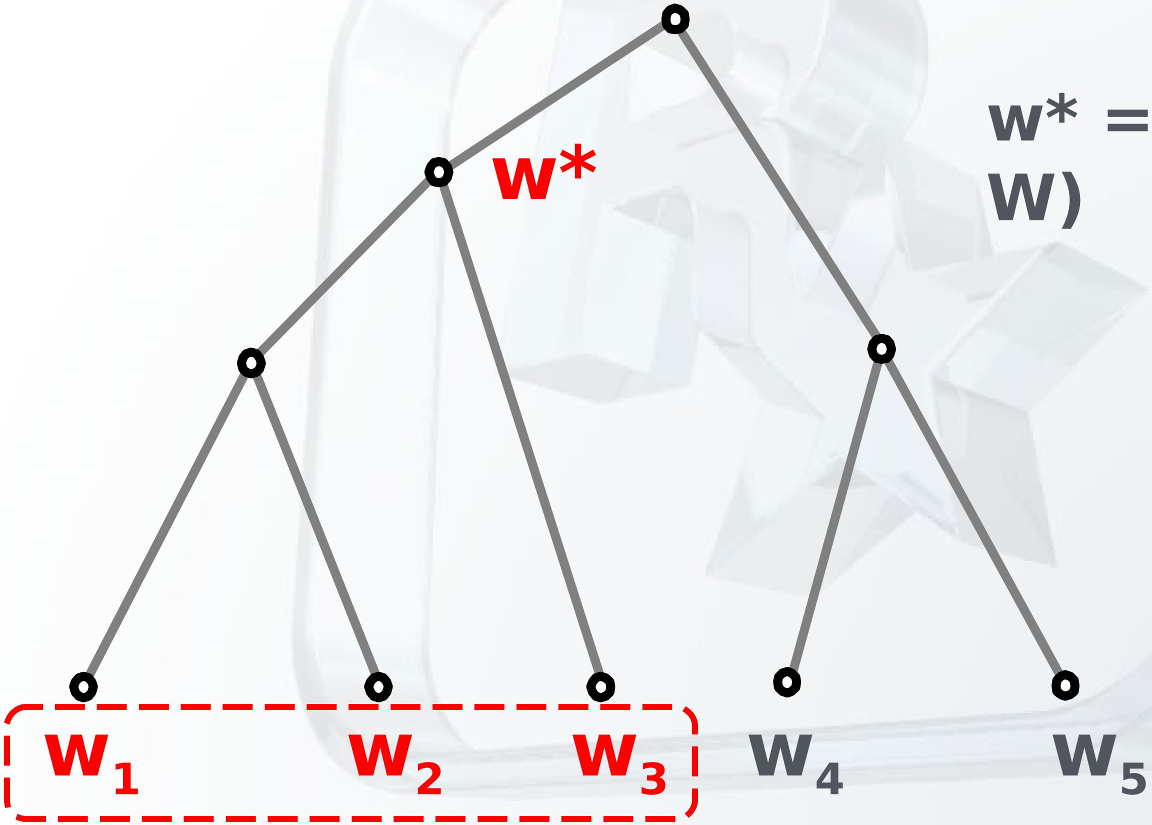
- **The target vertex v* of a contraction needs to get a weight w***
- **Repeated edge contractions create a vertex hierarchy.**

→ The weight of a root vertex should repre-sent all weights of the leaf-vertices

Weighted Error Metric(6)



Weighted Error Metric(6)



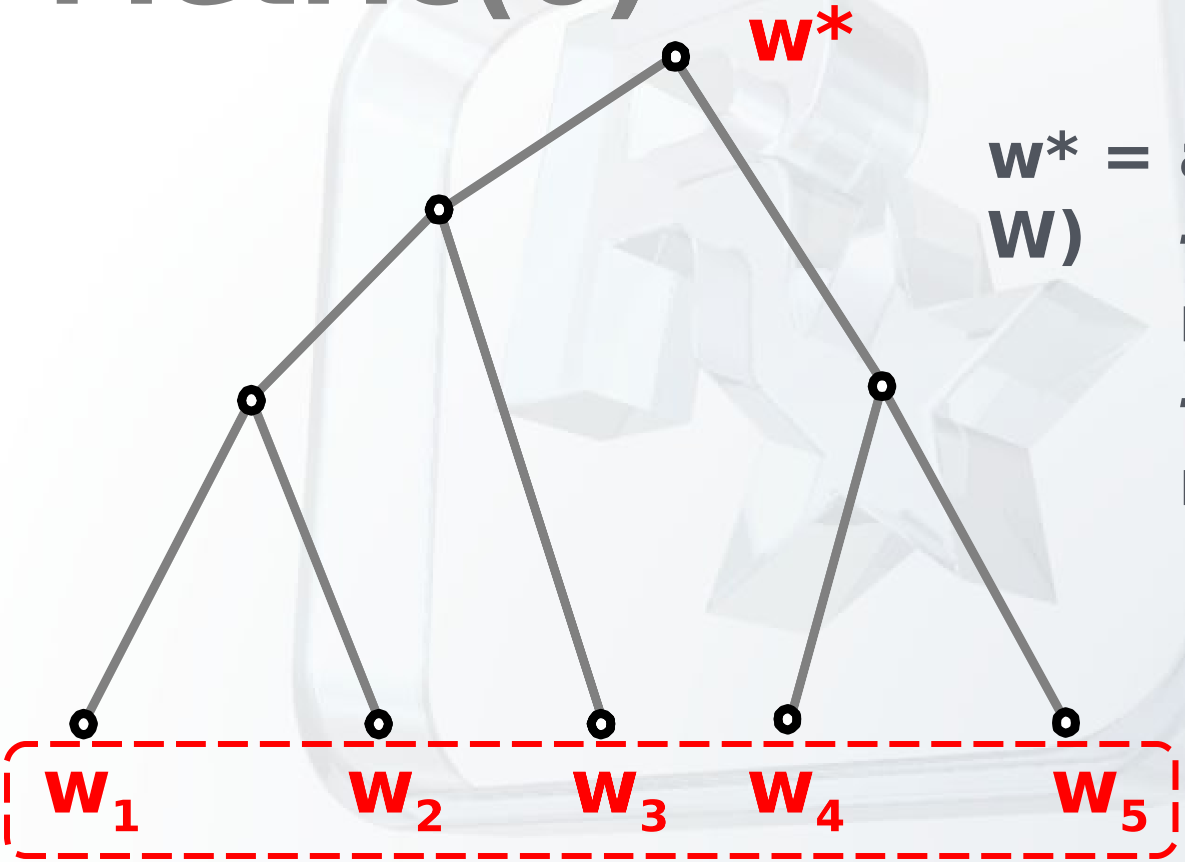
$w^* = \text{average}(w_i \in W)$ *- or -*

$\min(w_i \in W)$

- or -

$\max(w_i \in W)$

Weighted Error Metric(6)



$w^* = \text{average}(w_i \in W)$

- or -

$\text{min}(w_i \in W)$

- or -

$\text{max}(w_i \in W)$

User Interface(1)

Goals:

- **Easy to use, intuitive**
- **Integrated into modeling workflow**
- **Rich editing environment**

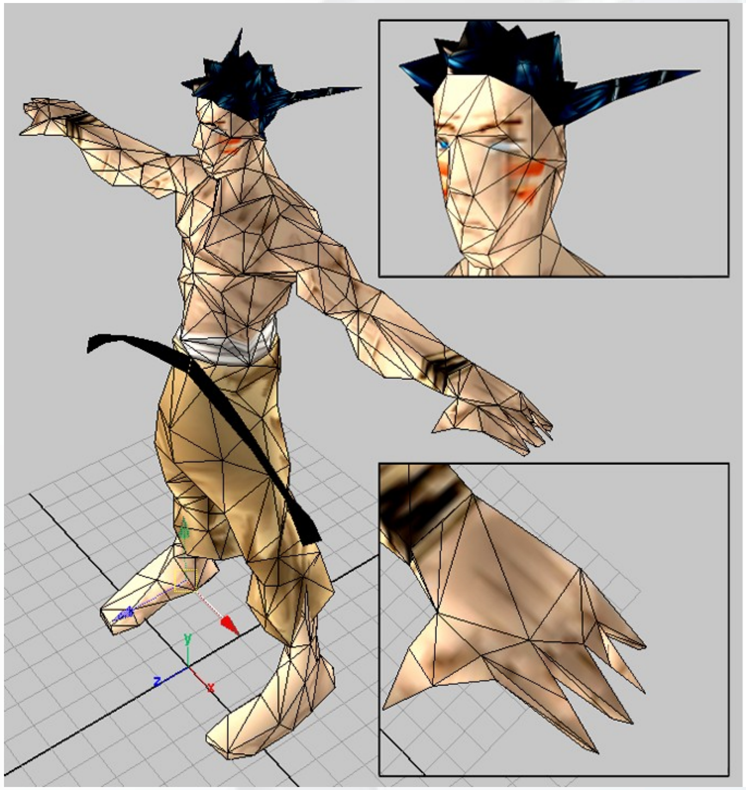
→ Sample implementation is released as Maya Plug-In

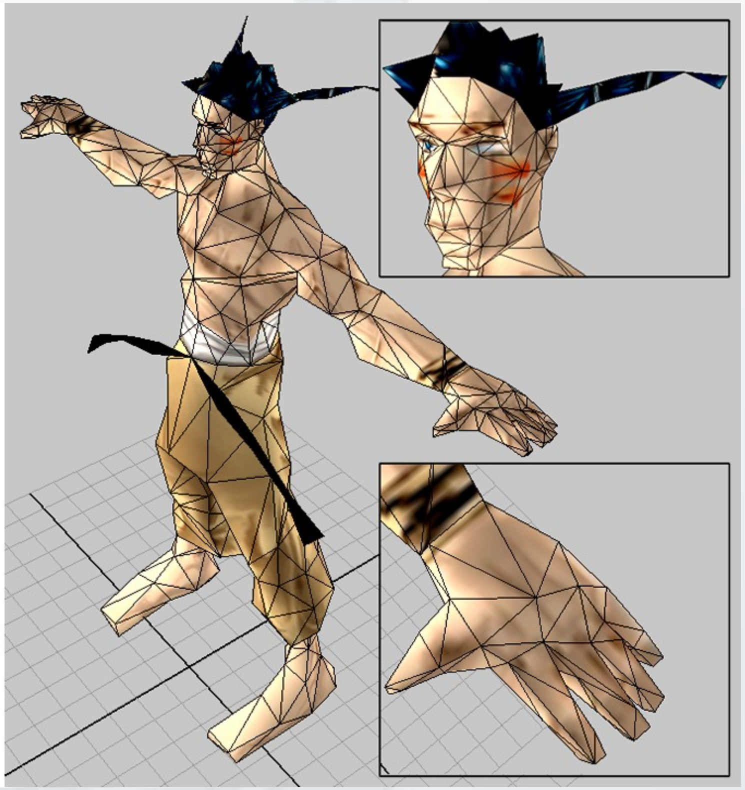
User Interface(2)

- Fully integrated into Maya
- Any Polymesh can be turned into a MRM
- Supports mesh attributes, including discontinuities
- Vertex weights are „painted“ using Artisan

Demo

Results(1)

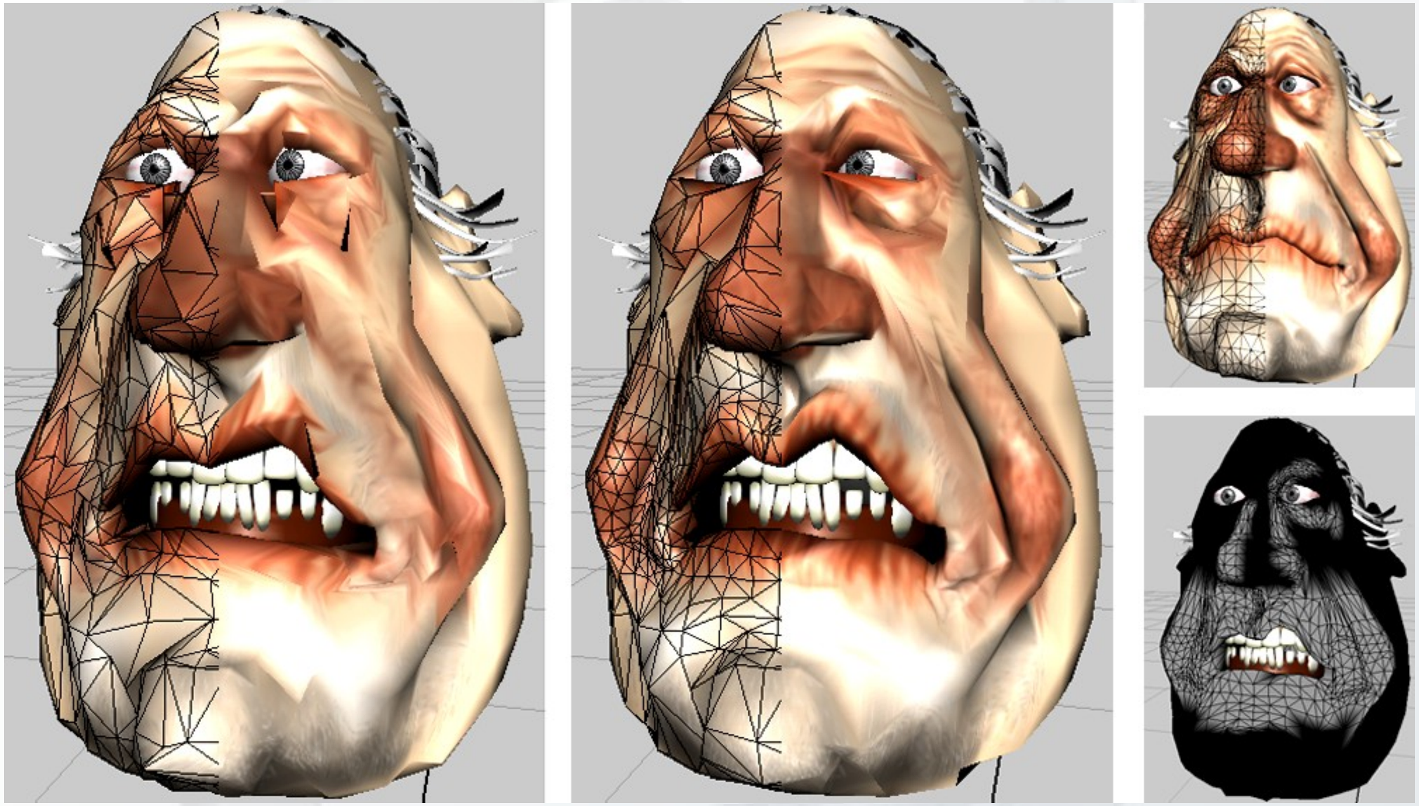




Reduced to 15%, original QEM

15%, weighted - Improved results

Results(2)



Facial animation: an example for functional importance

Conclusion

- User has control over simplification
- Weighted Metric is simple, but effective
- Can be applied to other metrics as well
- Release as Maya plug-in was a big success

Future Work

- Automatically try to deduce weights through animation sampling or texel density
- UI for additional constraints such as positional or border
- UI to exclude some parts of the model from simplification

Contact Information

Erik Pojar

erik.pojar@rockstarvienna.com

Dieter Schmalstieg dieter.schmalstieg@ims.tuwien.ac.at

Maya plug-in + source code:

www.pojar.net/ProgressiveMesh